Title

authors
CSAIL, MIT
authors@mit.edu

Abstract

abstract goes here.

1 Introduction

blah blah

2 Background

ML for programs. background

Graph neural networks. some more background

3 Approach

To solve the approximate type inference problem, we take a strong inductive bias over the types of predictors that we would expect to work on programs. Specifically, we take advantage of our knowledge of the important local and nonlocal interactions between various parts of programs, similar to how convolutional networks assume some strong relationship between local pixels (**?**). There are three specific priors we bake into our solution:

- **Nodes near a variable in the AST reflect something about that variable's type.** For instance, consider the following code snippet:

```
let x = y * 2;
```

A human trying to infer the type of the variable `x` in the above code may decide that since the result of multiplying something by a number is almost always a number, `x` is probably a number. This corresponds exactly to two types of edges that we include in our generated graph, an `AstChild` edge from parents to their child nodes in the AST, and an `AstParent` edge from children to their parent in the AST.

- **Nodes near a variable in the file reflect something about that variable's type.** Consider the following code snippet:

```
let x = 1;
let y = x;
...
let x = "foo";
```


Label ($n = 66,470$)	Vulnerable	Not vulnerable
`TOD`	2,360 (4%)	64,110 (96%)
`IntUn`	4,409 (7%)	62,061 (93%)
`IntOv`	18,544 (29%)	47,928 (71%)
`StateChange`	1,266 (2%)	65,204 (98%)

Table 1: Labels summary

It is clear that both ordering and locality matter in determining the type of `y`: in the AST, the two statements assigning to `x` are equidistant and directionally indistinguishable, but the actual *token stream* of file reflects that the first assignment to `x` precedes the assignment to `y`, meaning that it may be more likely to affect the type of `y`, and is also closer, meaning the two statements are more likely to be related. Because of this, we include two more types of edges in the induced graph, a `TokenNext` edge from a token to its neighbor in the source file, and a `TokenPrev` edge which is the reverse.

- **Nodes that use or define the same variable give information about each other.** Consider the following:

```
let x = 1;
...
let y = a + (b + (... + x));
```

Although the two statements may be arbitrarily nonlocal in both the AST and the source file, they are clearly related, in that `x`'s type probably influences `y`'s type. Because of this, we include two final types of edges, a `UseDef` edge from a variable's definition to its use, and a `UseUse` edge linking any two variable usages.

4 Method

5 Experiments

In this work, we investigate the following research questions

- How
- How
- Do features selected by models trained on complex representations provide a rich qualitative understanding?

We built a balanced set of roughly 66k functions and used blah blah

The dataset pertaining to each label was split into a training and test set in the ratio 66%-33%. The dataset for each

Label	N	Tr	Node	T	UD	UD+T	U	U+T	U+D+UD	U+D+UD+T
Tot # features		**2**	**44**	**79**	**8,232**	**8,311**	**14,940**	**15,019**	**23,172**	**23,251**
Avg # features trained on		**2**	**30**	**51**	**1,012**	**1,161**	**1,763**	**1,704**	**2,775**	**2,813**
TOD	**4,594**	0.47	0.72	0.70	0.78	0.80	0.76	0.79	0.80	0.80
StateChange	**2,440**	0.62	0.75	0.75	0.75	0.77	0.77	0.79	0.79	0.82
IntOv	**28,634**	0.66	0.78	0.81	0.82	0.84	0.84	0.86	0.85	0.87
IntUn	**6,676**	0.64	0.79	0.85	0.85	0.86	0.85	0.87	0.87	0.88

Table 2: Results summary. Test-set classification accuracies for ablation models.

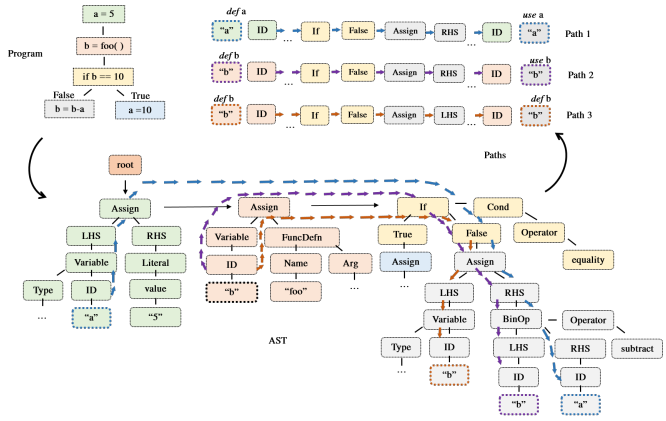

Figure 1: `Use & Define` Paths - An example. Paths to the expression $b = b - a$ are shown. Statements and their corresponding sub-trees have been shaded in the same color.

label was balanced - with roughly the same number of positive and negative samples, and stratified to ensure that the distribution of function types was similar in the train and test set.

Our models

- **Model1, `Tr`.** Oh boy.
- **Model2, `Node`.** We use

For models trained on our representations, we also analyze the highest contributing features. These two analyses help answer the three research questions we pose.

6 Results

The test-set classification accuracies of the different models we evaluated are tabulated in Table 2.

7 Related Work

Recent work have focused on .

8 Conclusion

This work presents an important first step in